Universitat de Lleida

TREBALL FINAL DE MÀSTER

Estudiant: **Josep Pascual Badia**

Titulació: Màster en Enginyeria Informàtica

Títol de Treball Final de Màster: Analysis of tourism in Catalonia using Twitter data

Director/a: Carles Mateu Piñol

Presentació

Mes: Setembre

Any: 2018

Analysis of tourism in Catalonia using Twitter data*

Carles Mateu and Josep Pascual

***Abstract*— This scientific paper is about analyzing the tourism in Catalonia with Twitter data. It details how to achieve a valid data analysis using building software and the validation of the data obtained through this software and the possibilities that this kind of tools could bring to the sector.**

I. INTRODUCTION

The main goal on this work is to realize a data analysis about tourism in Catalonia through the use of the public available data on social networks.

The purpose of this analysis would be a proof of concept to obtain tourism data in a more efficient way than the usual one, currently the Idescat (Catalan Institute of Statistics) obtains this information making surveys[1], this method has a material cost and the size of the sample studied is very limited in contrast to the number of incoming tourists, also you have to be careful in order to pick a relevant and representative part of the population as a sample.

Nowadays with the possibilities of the big data and combining it with the enormous growing potential that social networks have in number of users, is possible to take advantage of this situation and mine a dataset with the necessary information to create a cheaper and equally reliable data than the usual data sources.

For that reason we have been decided to use social media as a data source, since we can check the nationality of the users and follow the route of the places that they have followed in their trip. This is enough data to make a valid analysis. And giving us the opportunity to create more efficient tourism campaigns.

So for making a good proof of concept we're going to mine data from the social media analyze this data and check if it's possible to obtain a reliable survey.

II. DESIGN

A. Why Twitter?

The social media network that will be used in the data mining process will be Twitter, since is not the first social network in active users per month (which is Facebook with 2196 million active users according to Statistia) [3] is still one of the most popular ones (330 million active users) [4]. This quantity of active users is enough to acquire relevant data to make a valid analysis. Also Twitter offers a free API to the developers with enough methods to easily interact and retrieve the data that matters to us.

*This work was not supported by any organization

B. Architecture

The application will be divided in three parts:

1) Data Mining: The process of acquiring the datasets from Twitter.
2) Data Analysis: The process of transforming that data, and making the statistical analysis.
3) Data visualization: The process of getting the results of the analysis and showing them to the user in a proper approach.

1) Data Mining: This part would be the one used for acquiring the data from the source(Twitter), organizing and storing it to our database. The general idea is to connect to the Twitter Stream and obtain all tweets geo-located in Catalonia, when we obtain the tweet the next step is to analyze the user information about the tweet's owner in the following way.

After the analysis is done and the user origin is determined the process is responsible for storing the tweet, the user information and depending if the user origin is Catalan or not store the touristic information in another dataset. The tourist information that would be saved (in addition to the tourist country origin) would be the places visited by the tourist during the last three months until we find one that is form their own country.

2) Data Analysis: It would be 2 main parts of the data analysis the first one and where the real problem resides is the part exposed on the data mining process, which is to determine the user origin, that is because the location field from the twitter users is a plain text field, this is a controversial point because force us to use a third service as Google for example, or try to figure the user nationality using other data available on their profile.

The second part of the analysis is when we attack the database in order to retrieve the results of the data mining, doing counts, percentages and asking the right questions to the database.

3) Data Visualization: Finally the last part of the project is the responsible to show the data in a structured way to the user, the idea is to obtain the result of the database and generate some plots with the data in order to give to the users a quick idea about the general state of the tourism and also give to them the possibility to make a further analysis.

III. TECNOLOGIES

A. Python

The main language used to handle all the calls between the API's is Python[4]. In the application is used as the server side language, also is used to retrieve the data and manipulate it.

B. MongoDB

MongoDB[5] is a database used to store the information in JSON format, we use them to store our information because it has high performance options as map-reduce and also it maintains the same format of the API's (JSON objects) and Python dictionaries.

This database is a non-relational database so we can not handle the usual approach as we would have with the typical relational databases.

The way to interact with this database is through the PyMongo[6] library.

C. Plotly

Plotly[7] is the library that we use to generate the plots, charts and other graphs that we want to show to the user in order to visualize the data.

Plotly acts as a library but it has an API authentication method, because of their monetization model. For this project we will only use the free layer of the library.

D. API'S

1) Twitter API: The calls to the Twitter API[8] are made through the Tweepy[9] python library, basically it wraps all the methods and give them accessible with python. The library is used in our project to connect to the Twitter Stream and also retrieve user data and location data.

2) Google Geocoding API: The Google Geocoding API[10] is used in order to geo-locate the location place field of the users and be able to determine which is the origin of the tweet's author. Also is useful to build objects with the locations that we've been analyzing.

3) MapBox: MapBox[11] is another API that give us the possibility to build interactive maps with our data. We use it along Plotly to draw the data along the map of Catalonia and be able to show the touristic points in the map.

IV. IMPLEMENTATION

Since this project is only a proof of concept and it has any kind of budget it will be used only the free tiers of the technologies, furthermore the project has been optimized to run accomplishing this requirement.

A. Deployment

Our Python scripts will be running on Amazon cloud 9[12], which is a cloud IDE, so we can take advantage and run there our code, this limits our log space until 2GB.

The MongoDB will be running on mlab[13], mlab give us a free MongoDB database limited until 500 MB.

B. Acquiring the tweets

When acquiring the tweets we have to configure the stream in order to obtain the tweets inside a box with the coordinates of Catalonia, since this is a specification of the Twitter API the borders of Catalonia are not exactly defined, this will complicate a bit acquiring only the tweets that we want and not false positives . The right border is with the sea and the northern one with another country, so technically these two are the easiest ones to identify, since the one with the sea will stop giving us tweets by itself and the northern one when we acquire the tweet we can check the country code, the problem stands with the southern and western borders. A proposed solution to this problem could be to define exactly which are the coordinates of Catalonia in a complex figure and when acquiring a tweet check if the tweet coordinates are inside the figure. A not acceptable solution would be to update this places manually in the database, but it could be considered as a temporary fix.

C. Identifying tweets author origin

This is one of the most controversial parts of the project, since the field of location of the users is plain text field, so when a tweet from Catalonia is obtained we retrieve the author's data in order to analyze it, when the location field is obtained we can be in 3 cases:

1) The location also has coordinates. That's because it is a place validated by twitter, in that case we set that place as the real user origin. We take this case as the most reliable source of information. Because the user has taken the time to fulfill his profile with more information than the usual one.
2) The location field is only plain text. in this point the thing is try to geo-locate that text into a place using the Google Geocoding API, when we receive the response we check if it is a valid place, we consider only valid place a political place, which could be the personal addresses, all other types of addresses are more probably to be fake.
3) The location field is geo-located but it returns a non political address. When this is the case we analyze the data and try to determine a nationality for the user, then we validate this location with the nationality obtained with the data, if it matches then we assign the geo-located non political address to the user, otherwise we set the nationality given by the data.
4) The location field is not set. Then we don't have any other option than analyze the user data in order to determine the user nationality, and when it's determined we assign it to the user. If we are not able to determine a nationality the data is just discarded.

A fact that it has to be taken into account is that the people that set's Spain as their location cannot be trusted because a people living in Catalonia who has set Spain as his location could lead to a heavy increase of the tourism inside of the country. So when this case occurs we just analyze their data and determine the origin with the data, not the location field.

D. Determine origin using user data

When trying to determine user origin and we don't have any other option left that analyzing the data we consider the following data:

- Language: Actually we only check if the language is Catalan to automatically assign Catalonia to the author, but an improvement could be to check which

languages are only spoken in one country of the world to automatically assign that country.
- User tweet's location: We check all the tweets with a location from the user, and counting the countries we can determine which one is the most usual, so we suppose that this is the user country and we assign it to it as the origin.
- User friends location: This option is no implemented but it could be one of the best ways to correctly determine the user origin, it consists on check all the user's friends location and assign the most repeated one to that user. This option is not implemented because the necessary requests are limited by the API restrictions, however it would be possible to use it paying the premium layer of the API.

E. Retrieving touristic trip

Once the user origin is determined check if it is a tourist or no, tis trivial. Once a tourist is identified we retrieve all the tweets with location from the last three months and we keep them with the time-stamp until we reach the end or a location from their country.

After that a tourist is stored in the database with his origin and destinations.

F. Optimizing resources

Since we have reduced space and limited API calls we must minimize the disk usage and the calls to the Twitter API.

1) Storage: For doing this we set up a store manager in order to store only the useful data for us, first of all we are only going to save the tweets ID's when storing the tweets, if we need something else in a future we will be able to retrieve the entire tweet with only the ID. For the users usernames we will only store the username since it would be the same idea as the tweets, with the username we could retrieve anytime the full user information this will be saved on the logs of the python process and also the database for both tweets and users.

2) API Calls: In order to minimize also the API calls we first will check if the information that we receive it's already in the database, this is useful for the most active users that are always tweeting, in order to not re-analyze them and also for checking the locations. We've created a location database that acts as a Index, this way we can save the effort of analyzing plain text places and indexing them with the correct location in the database. This optimization is due to the call limits on the free layer of the Twitter API. As an example with this method will analyze the plain text Barcelona just one time, without this optimization we well ask the API to geo-locate Barcelona every time.

G. Database definition

The definition of the objects is this one:

1) Tweets: Here we store the tweet information, with the light storage we only store the tweet id (id), with full storage all the tweet object is stored.

2) Users: This collection contains the user information, like the tweets when the storage is set to light we only save the screen_name (id), because all the other data is available on Twitter.

Also we add the next fields:

- origin: A location object which give us the information about the user origin.
- data_determined: A flag to distinguish how we determined the user origin, using the label location or using the user data.

3) Location: In this collection we will store object representing the places that we can extract from Twitter. This object will have the next attributes:

- location(id): text field containing the plain text location.
- country_code: 2 digit country code.
- country: country name.
- catalunya: flag to know if the location is inside of Catalonia or not.
- geo: object with the properties, lon and lng representing the float longitude and latitude.
- address: The political standard address.

4) Tourists: These collection would be responsible for storing the tourist information, it will have the following fields:

- screen_name(id): user name of the tourist.
- origin: location object representing the tourist origin.
- destinations: A list of (location,time-stamp) objects which give us the possibility of giving an order to the destinations and be able to follow the tourists travel.

Having these collections defined and after a few weeks running the Python scripts, the only thing to do is attack the database and consume the data to present the results in a appropriate way to the users. Every collection has defined a Index by their own ID, in order to not duplicate the data and make the access more efficient.

H. Retrieving tourist information from the database

Once we have data in the collections previously defined we just have to retrieve it, in order to know how many tweets, users and tourists we have is just trivial, just making a count into the database will give us the result. Having this result we can divide the number of tourists between the number of users and multiply it for 100 and we will obtain which percentage of users are tourists.

```
def tweet_count(self):
  tweets = self.get_collection("tweets")
  return tweets.count()
```

Fig. 1. Python code that returns the number of analyzed tweets.

For counting how many tourist places exist for each country the thing to do is aggregate by tourist origin country and return the sum of this results.

For knowing how many times the tourists visited each country it becomes a bit more complex because we need to

```
query = tourists.aggregate([
  {"$match":{"destinations":{"$ne":[]}}},
  {"$group" :
    {
      "_id":"$origin.country_code",
      "count":{"$sum":1}
    }
  },
  {"$sort":{"count":-1}}
])
```

Fig. 2. MongoDB query that returns the different tourist origins and the quantity.

map each location value and reducing it summing all the values into the location name as the key.

```
tourists = self.get_collection("tourists")
map = Code("function(){"
  "this.destinations.forEach(function(z){"
    "if(z.location.catalunya==true){"
      "emit(z.location.location,1);"
    "}"
  "});"
"}")
reduce = Code("function(key,values){"
  "return Array.sum(values);"
"}")
result = tourists.map_reduce(
  map, reduce, "myresults"
)
```

Fig. 3. Map reduce that returns the routes of the tourists.

When we want to show the touristic travel we must retrieve all the destinations and link them in the order of the time-stamp that we previously saved into the database.

Now that we have these basic queries we can tune them and filter with our needs, for example if we want to know which places are visited only by certain country members we can tune the map reduce adding a filter composed with a variable that represents that country.

I. Representing the information

Once we receive the results we can't show them to the final users by terminal or using plots, so we must find the proper representation of this data.

A CLI application has been built in order to consume this data, but in future implementations of type-like projects this application can be substituted by a dashboard accessible through a web application.

With this interaction tools we can select the data that we want to see, for example we can retrieve the basic counts of the analyzed data.

The other options retrieve the data and using python scripts we adapt it into data structures to fulfill different kind of

```
tourists = self.get_collection("tourists")
map = Code("function(){"
  "if("
    "this.origin.country_code"
    "=='+country_code+"
  "'){"
    "this.destinations.forEach("
      "function(z){"
        "if(z.location.catalunya==true){"
          "emit(z.location.location,1);"
        "}"
      "}"
    ");"
  "}"
"}")
reduce = Code("function(key,values){"
  "return Array.sum(values);"
"}")
result = tourists.map_reduce(
  map, reduce, "myresults"
)
```

Fig. 4. Tuned map reduce that returns the routes of the tourists in a certain country filtered by the country code.

graphics that show the different countries of the tourists with typical plots as a pie plot and a bar plot to show the percentage and totals of each country per tourists.

Other options in order to represent this data but in a more graphical way, is to make a heat map with the countries on a world map.

We have also options for showing which tourists places are the most visited and the tourist origin distribution along these places.

Also we can retrieve the tourists on a region and create routes through their destinations in order to graphically see which places they visited before visiting Catalonia.

Also if we want we can track one user using their screen name to obtain his data and show the route that he followed.

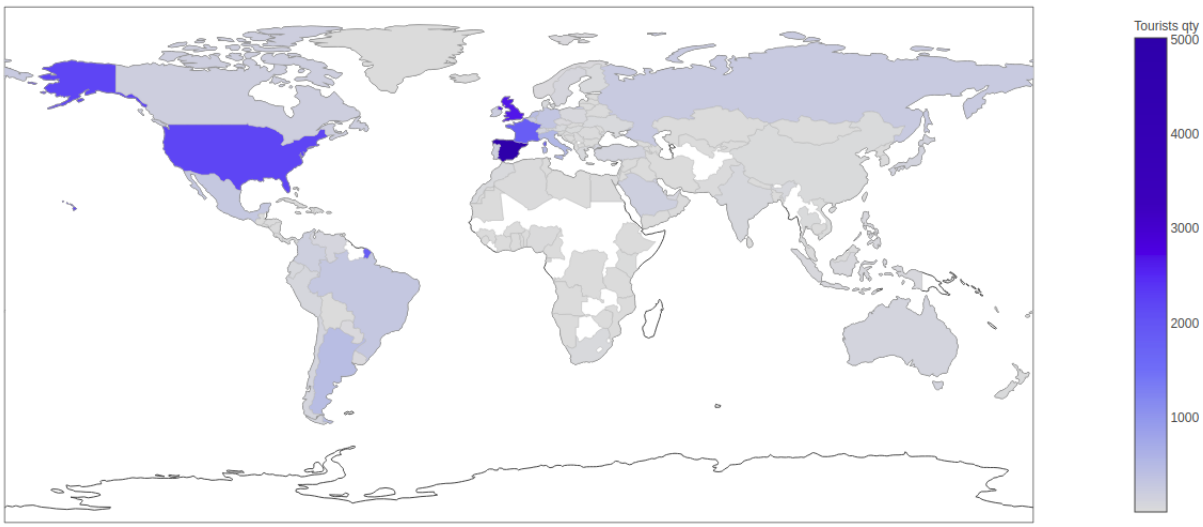

Fig. 5. Heat-map of the world with the tourist nationalities.

Fig. 6. Paths followed by the German tourists.

TABLE I
RESULTS COMPARISON

	Idescat(%)	TIS(%)	Difference
France	14.7	14.4	-0.3
United Kingdom	13.4	19.9	6.5
Rest of Europe	13.2	4.06	-9.14
Rest of the world	12.7	9.46	-3.24
USA	10.1	16.8	6.7
Germany	6.5	2.38	-4.12
Russia	5.6	2	-3.6
Norse countries	4.9	1.57	-3.33
Italy	4.8	4.2	-0.6
America (without USA)	4.8	15.5	10.7
Netherlands	3.2	3.54	0.34
Ireland	2.3	1.61	-0.69
Belgium	1.8	1.65	-0.15
Portugal	1	1.3	0.3
Switzerland	1	0.925	-0.075

TABLE II
STATISTICAL DIFFERENCE

	All	Aggregations	Individual
Max	10.7	10.7	6.7
Min	0.075	3.24	0.075
Avg	3.319	6.6025	2.125
Med	3.33	6.235	0.6

J. Validation

In order to validate the data that we acquired and classified into the different groups that we defined we're going to compare it against Idescat (Catalan Institute of Statistics)[14], the grade of versatility we will give to our data would be the grade of similarity that we will obtain when comparing the two datasets.

Observing the results we can observe that the greater differences are on the part of the aggregation so we are going to divide the results in two parts the differences between the aggregation and the individual countries.

We can see that for individual country model our process is really precise having the average difference on 2.12 points, and the overall result an average difference of 3.31. The difference between the individual countries and the aggregated countries could be due to the different estimation techniques that use the Idescat, and our system instead of estimations computed the numbers that we have on the database.

K. Improving the tool

Now that the project has reached the basic requirements we can think in improvements for this tool adding sustainable capabilities and also helping to retrieve better data.

1) Improvements: The data mining process can be adapted in order to consider every region of the world, filtering the tweets inside a complex geographic figure through the coordinates.

With the premium layer of the Twitter API we can increase the number and accuracy the users that we analyze making

more requests, for example for analyzing the friends of the user and also analyze the data of the Twitter trusted location users for re-validate this data with our algorithm, this could help to avoid intended fake data (troll users).

Also it can be made a touristic route map inside of Catalonia, treating the information in more complex but accurate maps.

Another option to consider is to create an interactive dashboard for the final user instead of the CLI application that is been developed.

2) New functionalities: One functionality we could add to this tool would be a place recommendation system, once we have the data of the users we can recommend similar places based on the data we mined with the tool.

Another capability we can add is to predict future campaigns using machine learning and AI systems.

Also it would be interesting to follow the international routes for the agencies in order to make international trip plannings.

V. CONCLUSIONS

After observing the result of the corresponding validation we can conclude that the difference is enough small to consider the data reliable, even if this difference some people think that is too big the process can be improved in order to reduce the gap. Of course the system is not perfect, but also are not the statistic estimations. The two methods have pros and cons, for example when you estimate data you are adapting models and also this models have a percentage of failure, on the other hand using twitter as a Source could bring some serious problems due to not all the population is on Twitter, and the sector that is on Twitter, is very significant in some ways (young people, high use of technologies, people providing fake data).

Also the economic budget is a decisive factor, so we should put in a balance which of the two models is the more optimum in relation price/quality.

Since the cost of this proof of concept is 0 it would be an interesting point to take into account for the tourism studies that are going to be realized, moreover, taking into account the proposed improvements for this project, tools like this have to be taken into serious account for the planning and optimization of the tourist campaigns on the becoming years.

REFERENCES

[1] Statistical planning. In Idescat Retrieved August 2018 https://www.idescat.cat/institut/idescat/memoria/2016/cap02.html

[2] eMarketer. 2017. Number of social media users worldwide from 2010 to 2021 (in billions). In Statistia Retrieved August 2018 https://www.statista.com/statistics/278414/number-of-worldwide-social-network-users/

[3] We Are Social; Kepios; SimilarWeb; TechCrunch; Apptopia; Fortune. 2018. Most famous social network sites worldwide as of July 2018, ranked by number of active users (in millions). In Statistia Retrieved August 2018 https://www.statista.com/statistics/272014/global-social-networks-ranked-by-number-of-users/

[4] Python. 2018. Python 3.7.0 documentation In python Retrieved August 2018 https://docs.python.org/3.7/

[5] MongoDB. 2018. MongoDB documentation. In MongoDB Retrieved August 2018 https://www.mongodb.com/

[6] MongoDB. 2018. PyMongo 3.7.1 Documentation. In MongoDB Retrieved August 2018 https://api.mongodb.com/python/3.7.1/

[7] Plotly. 2018. Plotly documentation In Plotly Retrieved August 2018 https://plot.ly/python/

[8] Twitter INC. 2018. Twitter developer platform. In Twitter Retrieved August 2018 https://developer.twitter.com/

[9] Tweepy. 2018. Tweepy documentation. In Tweeepy Retrieved August 2018 http://www.tweepy.org/

[10] Google. 2018. Google geocoding documentation In Google Retrieved August 2018 https://developers.google.com/maps/documentation/geocoding/start

[11] MapBox. 2018. MapBox documenation In MapBox Retrieved August 2018 https://www.mapbox.com/api-documentation/

[12] Clou9. 2018. Cloud9 documentation In Cloud9 Retrieved August 2018 https://docs.c9.io/docs

[13] mlab. 2018. mlab documentation. In mlab Retrieved August 2018 https://mlab.com/

[14] Foreign tourists by country of residence. In Idescat Retrieved August 2018 https://www.idescat.cat/indicadors/?id=conj&n=10306&lang=es